06

Sobreescritura

Transcripción

[00:00] Y si somos un poco más curiosos van a ver que el mismo nombre método aquí, getBonificación, es el mismo nombre de método aquí de la clase hija getBonificación. A esto se le llama sobreescritura de método. ¿Por qué sobreescritura?

[00:24] Porque estás usando la misma firma del método. Mira, copio esto aquí, vengo aquí, comento acá y copio aquí. Es exactamente la misma firma del método pero lo estoy repitiendo en la clase hija. A eso se le llama sobreescritura, yo estoy reescribiendo el método en la clase hija según su propia lógica.

[00:54] Ya en otros ejercicios de lo que van a ir practicando van a ver otros casos en los que pueden emplear la sobreescritura del método y van a ver por qué es tan útil esta característica. Ya nos va quedando un poco más claro el concepto de la herencia y cómo gracias a sus características propias podemos hacer incluso sobreescritura de métodos.

[01:18] Algo muy importante que quisiera que sepan es que esos conceptos de herencia que estamos viendo no solamente se aplican a lo que es Java. Si ustedes aprenden muy bien este concepto de herencia, de sobreescritura de métodos, extender de una clase, van a ver que se usa también en otros lenguajes de programación, por ejemplo C Sharp, Ruby, otros lenguajes orientados a objetos.

[01:45] También utilizan estos mismos conceptos. Por lo cual si entienden cómo funciona esto aquí, les va a servir para una infinidad de cosas, y entre esas infinidades otros lenguajes de programación van a poder explotar al máximo las características de otros lenguajes de programación.

[02:05] Vamos a suponer aquí ahora que cambió la regla del negocio y que el gerente ya de por sí ya gana su buen salario y de bonificación recibe otro buen salario y ahora va a recibir aún más, ahora va a recibir el 10% del salario. ¿Cómo calculo eso? Podría simplemente decirle getSalario + y a esto ¿qué le voy a sumar? Nuevamente super.getSalario por 0.1.

[02:47] Todo eso lo pongo en un paréntesis. Listo. Solamente para darle mayor visibilidad y le voy a dar un enter también para que se vea, para que entre en la pantalla. Perfecto. ¿Será que va a funcionar? Volvemos al TestGerente, le damos, aceptamos y 6600. ¿Por qué? Porque es el salario más el 10% del salario, 6600. Cumple con el objetivo.

[03:20] Y ahora quiero que vean otra cosa muy curiosa aquí y es esta línea de aquí. ¿No se les hace también conocida? Es exactamente esta línea de aquí, ¿cierto? Entonces, de igual forma que hemos hecho con el super, nosotros podíamos aquí también decirle que sea igual a super.getBonificacion, y el resultado de TestGerente, guardamos aquí, sigue siendo exactamente el mismo.

[03:57] ¿Qué es lo maravilloso de esto? Que si en algún momento, por ejemplo si yo hubiera tenido mi código como lo tuve anteriormente, así, ¿qué hubiera pasado si de un momento a otro, no sé, llegó la crisis o alguna situación muy inesperada y las bonificaciones tengan que bajar a la mitad? En este caso la bonificación llegaba a ser de 0.1, sino de 0.05.

[04:28] Perfecto. Y dado que el gerente también recibía bajo la misma condición, si yo me olvidé de actualizar aquí en la clase gerente, él va a seguir

ganando mucho más y eso va a dar errores de contabilidad y una inconsistencia de números en el sistema y va a ser todo un problema.

[04:54] Entonces una de las cosas que también me brinda la herencia es esa consistencia. Si el método aquí es consistente y es una sola fuente de la verdad, la comisión va a ser ahora 0.05. Automáticamente, si yo llamo al método de la clase padre, super.getBonificacion, entonces yo ya no necesito preocuparme por las clases hijas, porque ellas heredan todas las características de la clase padre.

[05:26] Recuerden eso. Voy a demostrarlo aquí, ejecuto y ahora es 6300 ya bajó la bonificación. Entonces mi código ya está quedando mucho más flexible, ya está quedando más organizado. Y como decimos también cuando evaluamos código ya está autoexplicativo. Ya el gerente a través del código ya da a conocer sus propias características y dado que tiene un extend, si yo deseo saber mucho más de él, llega aquí y observa las características propias de cada funcionario de la empresa.

[06:09] Entonces, ya hemos visto un poco más del uso de los conceptos de herencia, hemos visto lo que es la sobreescritura de métodos, les recomiendo mucha práctica y nos vemos en el siguiente video, chau.